Assignment 1.1
Web Services and Cloud-Based Systems

Brunink, Yannick
yannickbrunink@gmail.com

Mossinkoff, Rico
ricokoff@hotmail.com

April 7, 2020

Web Service (WDSL/SOAP)

This report discusses the development of a simple WSDL/SOAP web-calculator-service. Usually, there are two approaches to develop a SOAP service. These are the Bottom-Up and Top-Down approach. We have implemented both for this assignment. For both options (Bottom-Up and Top-Down) we have used Eclipse as our IDE, Tomcat as our server, and Maven for our framework. Also, we decided to use Java as our programming language. In the following sections we briefly discuss how we constructed both methods.

Bottom-Up

Eclipse has build-in packages to easily make web services. Apart from this, we imported some packages from the Java API for XML Web Services (JAX-WS). These packages can do a lot of the work to create the xml needed to make a SOAP service/client. We created a web service in Eclipse and implemented the calculator classes. We used maven to make it easier to import the right packages for our program. Using the @Webservice and @Webmethod annotations, we specified that this class should be used as a webservice. We used the RPC binding style. In the sun.jaxws.xml and web.xml files we declared some of the settings so the server knows how our service works. We specified the URL we want the service to run on and we gave some information about the servlet. After this configuration, we created a client using the server client package of Eclipse. This generated most of the files for a working client. We added some files to make sure the right packages were available to let the client run.

Top Down

The Top-Down approach was fairly simple. We used the wsdl of the Bottom-Up approach that we implemented. Then eclipse generated the necessary files for the Top-Down web service. We only implemented the generated functions and added some necessary files to make the service work without errors. The client was generated in the same way as for the Bottom-Up approach.

Question 3: Propose a solution to make the Calculator stateful (do not implement just describe the solution: text + diagram)

To make the calculator stateful, we can save a session for a user. Javax ws offers a build in package for this. In this session we can save previous calculations made by the user. This way, the user can use an "ans" functionality and perform new calculations with this. The history of calculations can be requested on demand by the user. Every user has his own calculation history. In Figure 1 a simple system design is depicted.

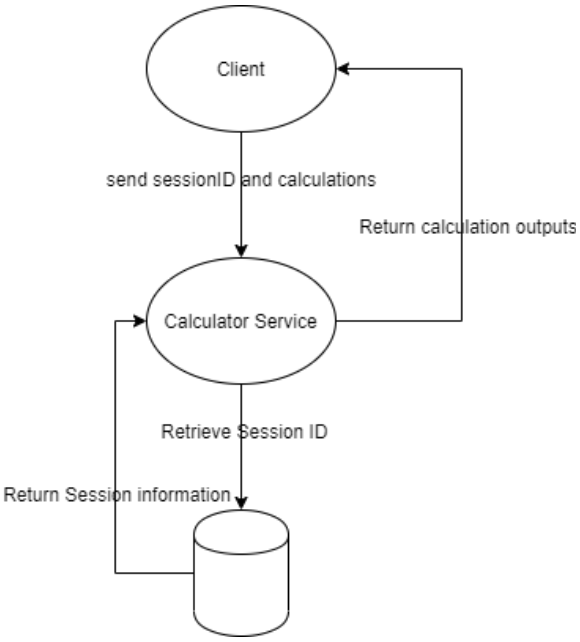

Figure 1: System design of SOAP with state